Regular Expression in Java (JAVA Regex) :-

- By edureka

Dt. 29/09/2023

Agenda

i) What is Regular exp?
ii) JAVA Regex.
iii) Matcher class.
iv) Pattern class.
v) Regex Character class.
vi) Regex quantifiers
vii) Regex metacharacters.

Regular Exp :- A regular expression is a sequence of characters that constructs a search pattern. When we search for data in a text, we can use this pattern to describe what we are looking for.

start of the line ↓ `^[a-zA-Z0-9_-]{3,15}$` ↑ end of the line

3 to 15 char long

Letters, numbers, underscores, hyphens

email/password validation

Regex Char. class :-

- ~~[a-x x-z]~~ `[abc]` → a, b, or c
- `[^abc]` → any char except a, b, c
- `[a-z]` → " b/w a to z
- `[A-Z]` → " b/w A to Z
- `[a-zA-Z]` → " a to z & A to Z
- `[0-9]` → " digit from 0 to 9

- `[a-d[m-p]]` → a through d or m through p (union)
- `[a-z&&[def]]` → d, e or f (intersection)
- `[a-z&&[^bc]]` → a through z except b or c (Subtraction)
- `[a-z&&[^m-p]]` → a to z and not m to p.

Quantifiers :- (repeatation)

- `[ ]?` → Occurs 0 or 1 time.
- `[ ]+` → " 1 or more times.
- `[ ]*` → " 0 or more " .
- `[ ]{n}` → " n times (n = 1, 2, 3 ...)
- `[ ]{n,}` → " n or more times.
- `[ ]{y,z}` → " atleast y times but less than z times.

Regex Metacharacters :- (shortform)

"\" tells comp to treat following characters as search character for '+'.
\ → escape character

\d → [0-9]

\D → [^0-9]

\w → [a-zA-Z_0-9]

\W → [^\w]

Q> 1> Mobile No. Start with 8 or 9 and total digit = 10

[89][0-9]{9}

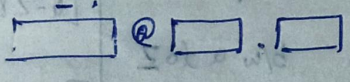
[89] → 1st digit

[0-9]{9} → any digit b/w (0-9), 9 times

2> First character uppercase, contains lower case alphabets, only one digit allowed in between.

[A-Z][a-z]*[0-9][a-z]*

3> Email id.



~~[a-zA-Z0-9]~~

[a-zA-Z0-9_\-\.]+[@][a-z]+[\.][a-z]{2,3}

\- → escape char

\. → escape char

JAVA Regex :-

The Java Regex is an API which is used to define a pattern for searching or manipulating Strings. It is widely used to define the constraint on Strings such as password and email validation.

Matcher Class :- Used to perform the match operation in the char seq.

i) boolean matches() → tests whether the given regex matches or not

ii) boolean find() → Used to find the next expression that matches the pattern

iii) boolean find (int start) → Searches the next expression from the given start number.

iv) String group() → Used to return the matched seq.

v) int start() → Returns the starting index.

vi) int end() → " " ending " .

vii) int groupCount() → " " total no. of the matche seq.

Pattern Class :- Compiled ver of a regular exp which is used to define the pattern for the regex engine.

i) **Static pattern compile(String regex) :-**
It compiles the given regex and returns the instance of a pattern.

ii) **Matcher matcher(CharSequence input) :-**
Used to create a matcher that matches the given i/p with pattern.

iii) **Static boolean matches (String regex) :-**
It works as a combination of compile and matcher methods.

iv) **String split[] :-**
Used to split the given String around matches of a given pattern.

v) **String pattern() :-**
Helps to return the regex pattern.

vi) **Int end() :-**
Returns the ending index.

Other regex metacharacters :-

. → any char
d → " digit
D → " non digit
s → white space char
S → non white space char
w → word char.
W → non word char
b → word boundary
Bb → non word boundary

Q) write a regular exp for ipv4 (A.B.C.D) where A/B/C/D digit limits b/w 0 to 255.

A.B.C.D

0-255

Let's break it,

0-9 → [0-9] → \\d

10-99 → [1-9][0-9] → [1-9]\\d

100-199 → 1[0-9][0-9] → 1\\d{2,3}

200-249 → 2[0-4][0-9]

250-255 → 2[0-5]{2,3}

String pattern = "(2[0-5]{2,3}|2[0-4]\\d|[01]?\\d\\d?)";

String ip = num + "." + num + "." + num + "." + num;